01 Dec 1993

A Simulated Annealing/Tabu Search Algorithm for the Vehicle Routing Problem

Jeffrey Dale White

Billy E. Gillett
Missouri University of Science and Technology

A Simulated Annealing/Tabu Search Algorithm for the Vehicle Routing Problem

J. White* and B. Gillett

CSC-93-35

Department of Computer Science
University of Missouri-Rolla
Rolla, MO 65401

*This report is substantially the M. S. thesis of the first author, completed December 1993

A SIMULATED ANNEALING/TABU SEARCH ALGORITHM FOR THE VEHICLE ROUTING PROBLEM

BY

JEFFREY DALE WHITE, 1968-

A THESIS

Presented to the Faculty of the Graduate School of the

UNIVERSITY OF MISSOURI - ROLLA

in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

in

COMPUTER SCIENCE

1993

Billy E. Gillett, Advisor

Fikret Ercal

Lee J. Bain

ABSTRACT

The Vehicle Routing Problem is an NP-complete problem that has been studied extensively since it was introduced in 1958 by G. B. Dantzig and J. H. Ramser. This thesis creates three algorithms that endeavor to find an optimal solution for each problem tested. Two of the algorithms (Simulated Annealing and Tabu Search) have been used previously to solve this problem. These two solution methods are revisited to discover whether a new approach to creating routes will produce the best-known optimal values every time. New routes are created by forming route neighborhoods and then selecting cities from these neighborhoods for insertion. The third algorithm is an original algorithm which combines Simulated Annealing and Tabu Search. The algorithms presented do not produce the best-known optimal values, but are competitive with previously published algorithms.

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

LIST OF ABBREVIATIONS

OSA . . . . . Osman's Simulated Annealing

OTS . . . . . Osman's Tabu Search

PF . . . . . Pureza and França Tabu Search

SA/TABU . . . . . Simulated Annealing/Tabu Search

T . . . . . Taillard's Tabu Search

TR . . . . . Gendreau, Hertz, and Laporte Tabu Search

VRP . . . . . Vehicle Routing Problem

WSA . . . . . White's Simulated Annealing

WTS . . . . . White's Tabu Search

I. INTRODUCTION

Efficient routing of vehicles has been and still is a major concern for trucking companies. If company A can use its trucks more efficiently than company B, then company A will not have to expend as many resources as company B. If company A is able to reduce its costs in this manner, then company A may have a better chance of survival than its competitor.

Vehicle Routing plays a major role in this scenario. It is known that there are non-zero costs (maintenance and operational) coupled with any piece of equipment purchased. If the costs associated with a piece of equipment can be minimized, then the resources to be used for those costs can be redistributed to other company functions. One way to minimize costs is through efficient routing of vehicles.

The Vehicle Routing Problem (VRP) can be formulated as follows:
Let $C = \{c_0, c_1, ..., c_n\}$ be the cities to route with n being the number of cities. Let c_0 be the depot for C. At c_0 are housed m trucks with capacity K, where $m << n$. At each city c_i where $i = 1, 2, ..., n$ there exists a non-zero demand q_i such that $q_i \leq K$. A truck must visit a city exactly once and every city must reside on one and only one route. Every route must begin and end at the depot. The problem focus is to route the trucks in such a way that the distance covered by the total fleet is minimized. In other words:

$$\textit{Minimize Cost}(R) = \sum_{p=1}^{m} \sum_{(c_i,c_j)\in R_p} d(c_i,c_j)$$

such that every demand is met and no truck capacity is exceeded. (c_i, c_j) is an arc in route R_p, $d(c_i, c_j)$ is the distance between c_i and c_j, and R is the total route configuration.

The objective of this thesis is to introduce and compare three solution methods of the VRP. The three solution methods to be discussed are Simulated Annealing, Tabu Search, and a hybrid (Simulated Annealing/Tabu Search). The first two solution methods have been investigated by other researchers, performing admirably and in some cases, finding the best-known optimal values for certain problems. The third solution method is a new algorithm. It uses Tabu Search to guide Simulated Annealing.

Section II of this thesis presents a review of the literature. A detailed discussion of the three algorithms is presented in Section III. Results are given in Section IV. Section V contains concluding remarks and future research opportunities.

II. REVIEW OF THE LITERATURE

The Vehicle Routing Problem (VRP) was first posed with solution by G. B. Dantzig and J. H. Ramser in 1958 as a generalization of the Traveling Salesman Problem. Their specific problem dealt with dispatching fuel trucks from a depot such that both the number of trucks and the total distance traversed by the trucks were minimums [DaR 59]. Their solution method creates tours in stages, specifically stages of groupings [DaR 59]. Each stage creates a tour that does not exceed a fraction of the truck capacity. The fraction used in their method is $[(\frac{1}{2})^i]K$ where i ranges from N, N-1,..., 0 and $N = \log_2 t$ where t is the maximum number of stops any one truck can make given its current capacity K. These subtours are optimized before proceeding to the next stage. This continues until tours with $[(\frac{1}{2})^0]K$ or less capacity are reached. Their solution method for the fuel truck problem produces a suboptimal route 1.4% larger than the optimal [DaR 59].

This research has spawned many versions of the VRP. Some problems currently being researched are the Fleet Size and Mix Problem, Vehicle Routing and Scheduling with Time Windows, Multiple-Depot Vehicle Routing, and Stochastic Vehicle Routing. The Fleet Size and Mix Problem does not use homogeneous trucks like the VRP. The trucks in this problem each have a different maintenance cost and capacity. In Vehicle Routing and Scheduling with Time Windows the trucks must meet the demands of

customers as in the VRP, but their customers are only able to accept deliveries during certain times of the day or on certain days. The Multiple-Depot Vehicle Routing Problem, in contrast to the VRP, contains more than one depot from which trucks are to be routed. The Stochastic Vehicle Routing Problem entails the use of probabilistic customer demands. One stipulation in this problem is that the trucks must be routed before knowing any of the demands. A penalty is assessed for any truck not able to completely service its route.

Much literature has been composed about different solutions to the VRP. Solution methods include branch-and-bound [Chr 69], savings [ClW 64], 3-optimal tour [Chr 69], man-machine interaction [Kro 72], set partitioning [Chr 69], dynamic programming [Chr 69], direct tree search [Chr 69], generalized Traveling Salesman Problem [Chr 69], sweep [GiM 74], and cyclic transfer [ThP 89]. Two other solution methods of particular interest are Simulated Annealing and Tabu Search. Both have been applied to the VRP by Osman [Osm 93].

Simulated Annealing and Tabu Search as implemented by Osman [Osm 93] construct the routes according to the Clarke-Wright savings algorithm. For Simulated Annealing a test run is performed, gathering parameters that will dictate the starting temperature, ending temperature, and temperature decrement calculation factors. For Tabu Search a test run is performed to initialize the tabu list. This Tabu Search uses only two components (tabu status and aspiration criteria) of the four Tabu Search characteristics [Glo 90].

Both algorithms use two interesting processes to obtain new routes: a shift and an interchange. The shift process simply moves a city from one route to another route. The interchange process exchanges cities between routes. Each city from route A will be exchanged with a city in route B as long as the two truck capacities are not exceeded. After these processes have been performed, a 2-optimal Traveling Salesman procedure is employed to optimize the new routes [Osm 93].

Tabu Search has also been applied to the VRP by Gendreau, Hertz, and Laporte [Gen 92]. The Tabu Search introduced by Gendreau, Hertz, and Laporte constructs routes based on an algorithm called GENI (Generalized Insertion Routine) [Gen 92] which inserts a city into a route between two cities c_i and c_k. A fascinating aspect of this algorithm is that cities c_i and c_k do not have to be adjacent when the city c_t is inserted, but they both will be adjacent to c_t after the insertion. This algorithm also uses the US (Unstringing/Stringing) routine to route the cities [Gen 92]. This procedure performs the operation implied by its name. At some point in the solution method the routes will be "unstrung" and then "restrung" into a better route configuration. This solution method uses all of the Tabu Search characteristics (tabu status, aspiration criteria, diversification, and intensification) [Glo 90] when deriving any solution, optimal or suboptimal [Gen 92]. The results from this algorithm appear promising. Of the fourteen problems investigated the known optimal values were found for nine of them. The other five values were within 0.8% of the known optimal values.

The VRP has been and still is extensively studied. As has been stated before, various solution methods have been applied to this problem. The results gathered from these solution methods have been mixed at best. The purpose of this paper is to revisit two solution methods already used on the VRP and to introduce a new algorithm created from existing solution methods.

III. STOCHASTIC ALGORITHMS

A. GENERAL ALGORITHM

Each of the algorithms used to solve the VRP will be discussed. However, the base structure of the three algorithms is the same. The only difference exists in their solution method. The general algorithm is detailed in Figure 1.

```
Begin
  Input truck capacity, location of depot, location of cities, and demands.
  Input parameters for SA, TABU, or SA/TABU.
  Create upper triangular distance matrix.
  Create initial routes using the "sweep" technique starting at city i.
  j = city i.
  Create neighborhoods for routes.
  z = 1st city in initial route #2.
  While (j ≠ z)
    Begin
      Solve problem using SA, TABU, or SA/TABU algorithm.
      Print out solutions to file i.
      i = i + 1.
      Move to city i.
      j = city i.
      Create next set of initial routes using the "sweep" technique
        starting at city i.
      Create route neighborhoods.
    End.
End.
```

Figure 1. General VRP Algorithm

An upper triangular distance matrix is implemented with linked lists in an effort to save memory since the VRP being solved uses symmetric distances between cities, i.e.

$d(c_i, c_j) = d(c_j, c_i)$. The "sweep" technique will be used to find an initial route configuration and then this configuration will be passed to the current solution method [GiM 74]. The solution method will then try to produce the best known optimal value for a problem.

A portion of the "sweep" algorithm created by Gillett and Miller [GiM 74] is used to generate an initial solution. The initial solution is created by first sorting the cities in increasing polar coordinate angle order. The first city derived from this sorting is chosen as the first city on route 1. The cities are "swept" for route 1 until the truck's capacity has been exceeded. The route is then returned to the depot. From the city where the truck capacity was exceeded, the next route starts. These steps are repeated until every city is routed. This initial solution is then passed to either Simulated Annealing, Tabu Search, or the Simulated Annealing/Tabu Search hybrid for solution improvement.

Once the initial solution is generated, solution space exploration commences by choosing route neighbors to insert into and remove from routes creating new route configurations. Unlike previously published algorithms, the three algorithms presented in this thesis use the concept of route neighborhoods. The TR algorithm created by Gendreau, Hertz, and Laporte, for instance, implements p-neighborhoods [Gen 92].

In the TR algorithm q cities are randomly selected for insertion. Each of the q cities is surrounded by a p-neighborhood where p represents the number of neighbors allowed in each neighborhood [Gen 92]. Each neighborhood is composed of the cities that are closest to each of the q cities. To insert a city into a new route, one of its p-

neighbors must be in the route. If the route contains a p-neighbor, a search commences locating the best insertion. This insertion might not occur between adjacent cities, but after the insertion is complete those cities will be adjacent to the newly inserted city. After the insertion, the resulting route is optimized. This insertion and optimization procedure is known as the Generalized Insertion (GENI) routine. At some later time, every route will be reoptimized using the Unstringing/Stringing (US) routine. This routine will use GENI to optimize every route. In effect US "unstrings" the route and then "restrings" it.

The route neighborhood concept is a unique way of looking at reducing the cost of a route. Figure 2 illustrates the beginning and ending angles of routes that aid in

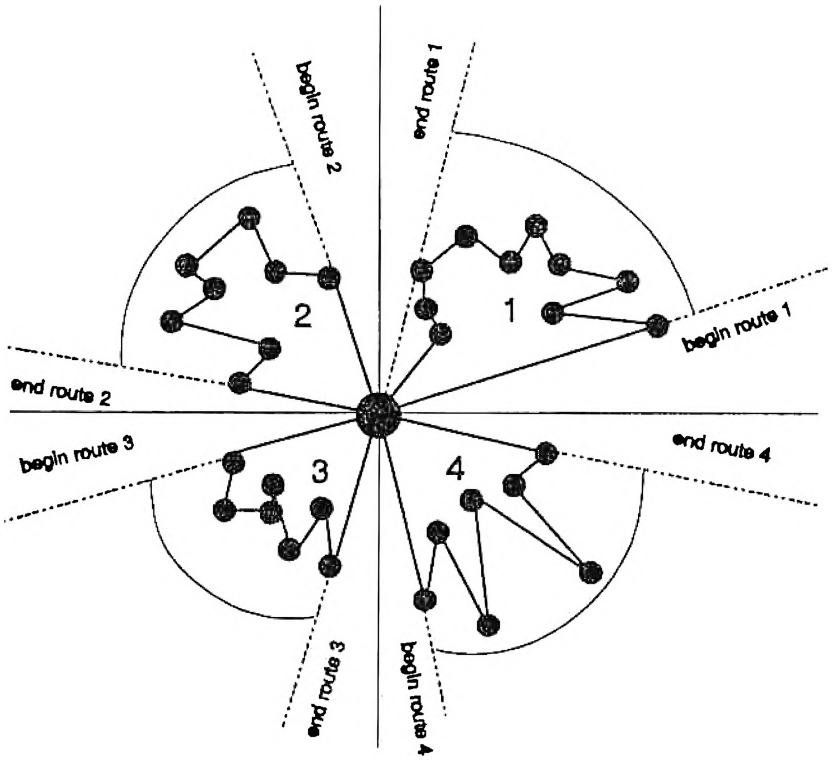

Figure 2. Beginning and Ending Example Routes

locating route neighborhoods. Every route has a set of neighbors. These neighbors are chosen by moving either a $\pm\delta$ angle from a bisector. The bisector is the line that divides in half the angle made between the beginning angle of route j and the ending angle of route i.

Suppose that Figure 2 represents an initial routing. The dashed lines represent the beginning and ending of each route. Figure 3 shows the bisector of each begin-end pair represented by γ_i (i=1, 2, 3, 4).

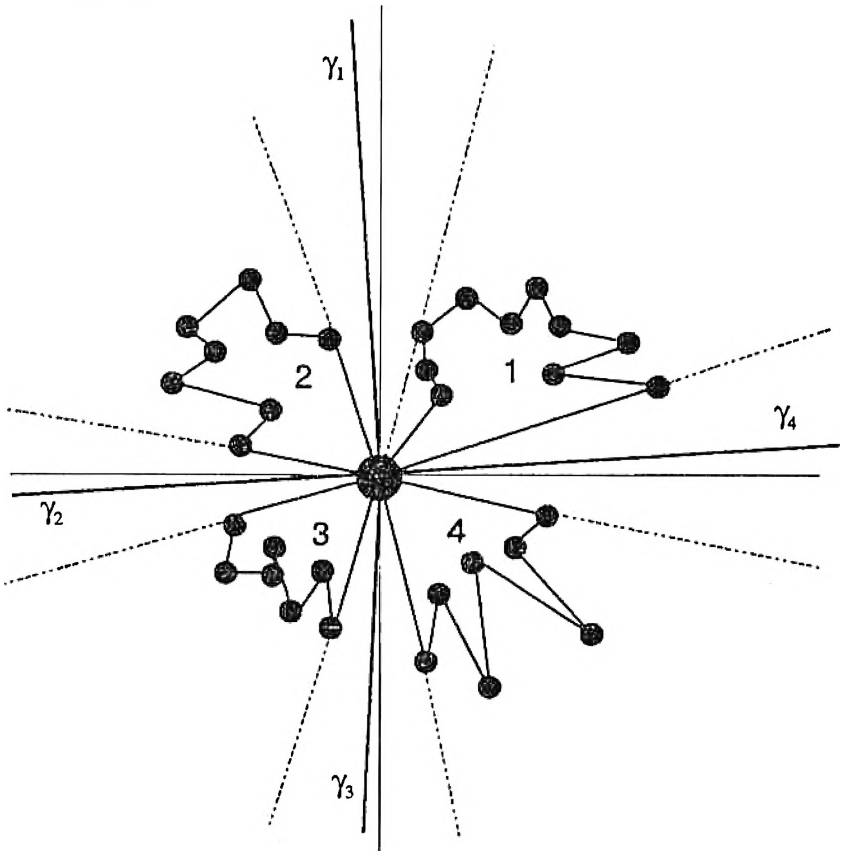

Figure 3. Bisector between routes

From these γ_i a fixed angle δ is "swept" either clockwise or counterclockwise, depending upon the boundary for which neighbors are being located. Any city falling

within either the range $[\gamma_i-\delta, \gamma_i]$ for beginning neighborhoods of route i+1 or $[\gamma_i, \gamma_i+\delta]$ for ending neighborhoods of route i is considered a neighbor of that route. For the case i=m, where m is the number of trucks, $[\gamma_i-\delta, \gamma_i]$ will be used to find beginning neighbors for route 1. These neighbors are stored so that they can be used when finding a new route configuration.

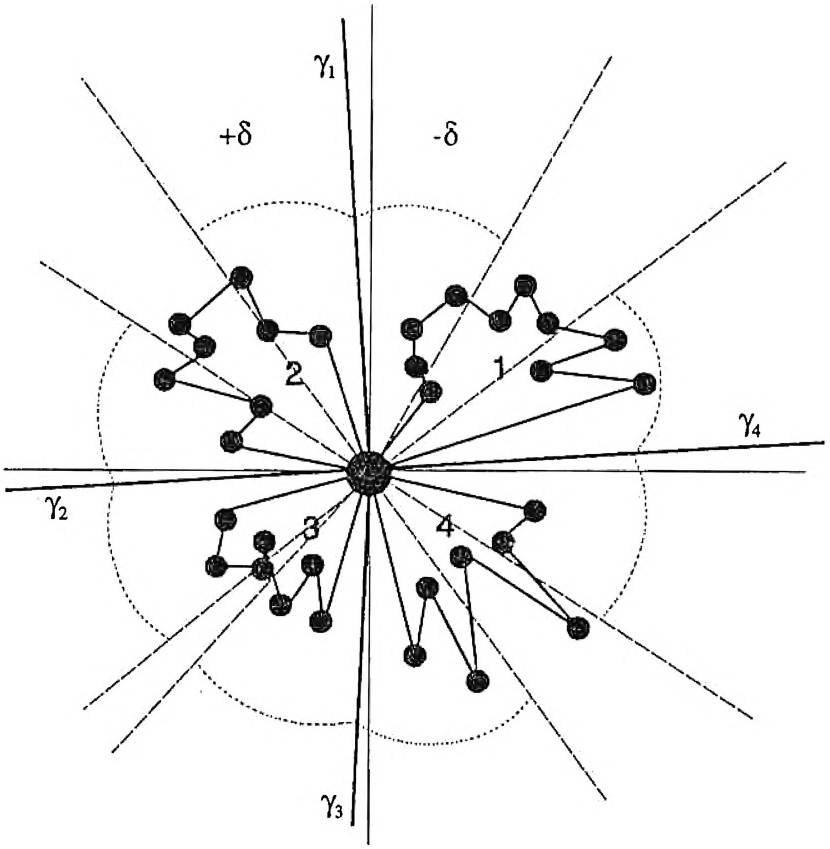

Figure 4. ±δ Neighborhoods

Each route has two neighborhoods, a beginning and an ending neighborhood as depicted in Figure 4. From these neighborhoods are selected the cities that will be inserted into the route. Every time a neighbor is chosen to be inserted into a new

route and that configuration is accepted, the neighbor is moved to a new neighborhood. The neighbor is moved to a new neighborhood because a city in route i can be a neighbor of route j, but not a neighbor of route i. For instance, if route j's neighborhood includes city c_t and c_t is moved into route j, c_t can no longer be a neighbor of route j. Thus, c_t must be moved to a new neighborhood of either route k (succeeding route) or route i (preceding route) depending upon which of route j's two neighborhood's c_t was extracted. These neighborhoods are constantly changing because of the insertion and removal of cities into/from routes.

After a solution method has obtained a new routing, the Lin 3-optimal Traveling Salesman procedure is applied to each route [Lin 65]. This procedure is well-known in combinatorial optimization for its use in solving the Traveling Salesman Problem. This algorithm finds the best possible Traveling Salesman tour connecting a set of points by exchanging three route links with three other route links, keeping the route a tour after the route links have been exchanged. This algorithm performs at most $\binom{n}{3}$ exchanges, where n is the number of points in the current tour.

B. SIMULATED ANNEALING

The first heuristic algorithm applied to the Vehicle Routing Problem was Simulated Annealing. This algorithm mimics the actual process of physical annealing [Kir 83]. The Simulated Annealing algorithm first raises the system to a "melt." This "melt" is then "cooled" by reducing the temperature in slow stages until no changes in the system are accepted. The simulation must be allowed to proceed long enough at

the current temperature enabling the system to reach a steady state or else solutions produced may be suboptimal at best. Simulated Annealing allows the system to deteriorate by accepting non-improving configurations. These configurations will be accepted probabilistically according to the e^{-b} factor, where $b = \Delta C/T$. The fraction, $\Delta C/T$, represents the difference between the current and new configuration divided by the current temperature of the system. A general Simulated Annealing routine follows the general outline shown in Figure 5 [Kir 83].

```
Begin
  Temperature = Initial Temperature.
  Iterations = 0.
  While (outer loop stopping criterion is not satisfied)
    Begin
      While (inner loop stopping criterion is not satisfied)
        Begin
          New Solution = GENERATE(current solution).
          If (accept(new solution, current solution, temperature))
            then current solution = new solution.
          Decrement temperature.
        End
    End
End of Simulated Annealing.
```

Figure 5. Simulated Annealing Algorithm

For the Simulated Annealing algorithm used to solve the VRP the outer loop stopping criterion is the rejection of every new configuration. The inner loop stopping criterion is the length of the Markov Chain. The Markov Chain length specifies the number of new configurations the algorithm should generate for a given temperature. The GENERATE routine, the heart of every Simulated Annealing program, generates

the "moves" to be used by Simulated Annealing. A "move" is generated by visiting each neighborhood and possibly selecting a neighbor to insert into a route.

The GENERATE routine works in the following way. For each neighborhood, a decision variable is set to either 0 or 1 with equal probability. If the variable is set to 1, a city will be selected from all cities in the neighborhood with equal probability for possible insertion into another route; otherwise, the routine moves to the next neighborhood. The result of these choices will be an array of size 2N where N is the number of trucks needed to cover the demands in the current routing. Each "move" is checked for feasibility before the cities are used to generate a new routing. The feasibility check reveals if any "move" will create an overweight truck. If this check allows the "move", the "move" will be passed to Simulated Annealing. If the "move" is infeasible or the array contains no cities, GENERATE will create a new "move".

This feasible "move" is then applied to the current routes. The new routing is checked against the current route configuration to see if the new routing costs less than the best configuration so far. If the new set of routes costs less than the current set, the current configuration is discarded and the new configuration becomes the current one. If the new routing does not cost less than the current routing then the new configuration is accepted probabilistically, according to the following factor:

$$e^{\frac{-(Cost(R_{new})-Cost(R_{curr}))}{Temp}}.$$

A clarification is in order. In the exponential function, a cost function, Cost( ), is shown. The cost function used for this problem was the configuration's total route distance. Since no trucks are allowed to exceed capacity and every truck has no limit

on distance, the logical choice for the cost function was total distance. Another reason it was chosen as the basis for the cost function was because the distance or cost of each route was easily accessible.

To aid in faster convergence to optimality, a check is applied to find the best location in which to place a city into a new route. This check is performed by finding the closest neighbor (in distance) to the city being inserted. Once the closest neighbor is located, a search from that neighbor backward two cities and forward two cities ensues. This search finds the best possible insertion. A best possible insertion is defined to be an insertion which adds the least distance to the current route. The search is performed from z-2 to z+2, where z is the closest neighbor to p, and p represents the point to be inserted. This search looks at each insertion between pairs of points (z-2, z-1), (z-1, z), (z, z+1), and (z+1, z+2). Once the best possible insertion is located, p will be inserted into the route. Every insertion is handled in this fashion. Figure 6 illustrates this insertion method.

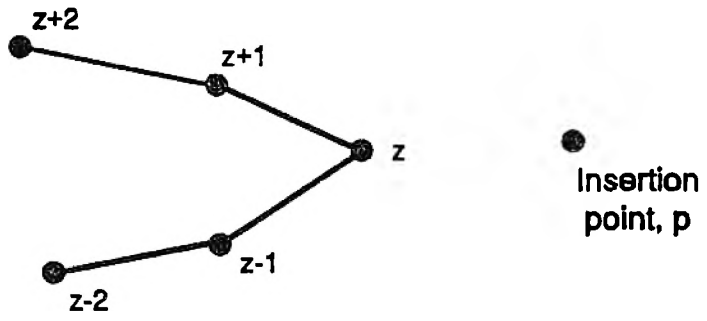

Figure 6. Insertion Example

C. TABU SEARCH

Another solution method for the Vehicle Routing Problem entailed using the Tabu Search heuristic. Tabu Search has recently been used to solve hard (NP-complete) combinatorial problems. For certain classes of problems, this search heuristic has performed better than present-day heuristics, most notably Simulated Annealing. It has even been heralded "by the Committee on the Next Decade of Operations Research (1988) as 'extremely promising' for future treatment of practical applications" [Glo 91].

Tabu Search combines four elements to solve combinatorial problems or to guide other solution methods. The four elements are short term memory, aspiration criteria, intensification, and diversification [Glo 90]. Although these four elements help to compose a complete Tabu Search, the first two can be used alone. Tabu Search, as presented in this thesis, uses the first two elements in its search for the best known global optimal values for the Vehicle Routing Problem.

The Tabu Search algorithm for solving the VRP studied in this paper is shown in Figure 7. This algorithm does not appear to be using the short term memory or aspiration criteria, but it is. These two Tabu Search characteristics reside in the GENERATE_NEW_ROUTING routine. The GENERATE_NEW_ROUTING routine is similar to the GENERATE routine in the Simulated Annealing algorithm, but before the "move" is executed on the current route, the "move" is executed on a test route, a copy of the current route. After the "move" is applied to this test route, the configuration cost (route distance) is calculated. The "move" is then checked against

```
Begin
  Best solution = sum_of_routing(current routes).
  Rejections = 0.
  While (rejections < maxrejections)
    Begin
      New routes = GENERATE_NEW_ROUTING(current routes, current
        neighborhoods)
      New solution = sum_of_routing(new routes).
      If (new solution < best solution)
        Begin
          Best solution = new solution.
          Save_best_routing(new routes, best routes).
          Rejections = 0.
        End
      Else
        Rejections = rejections + 1.
      Current routes = new routes.
      Update_neighborhoods(current neighborhoods).
    End.
    Return best routes.
End of Tabu Search.
```

Figure 7. Tabu Search Algorithm

the tabu list. While this check of the tabu list is occurring, the configuration cost and truck capacities are checked. If the "move" is tabu or forbidden but the configuration cost is better than the best "move" encountered thus far and the trucks are not over capacity, the "move" is accepted as the "best" current move. One other important difference between GENERATE and GENERATE_NEW_ROUTING is that GENERATE_NEW_ROUTING will generate M "moves" and then pick the "best move" out of the M "moves." This enables Tabu Search to aggressively search for the global optimal value or a local optimal value. One reason that Tabu Search was chosen to solve this problem was its ability to "remember" previous "moves." This

"memory" capability discourages another visit to a "move" until t iterations have passed, where t is the length of the tabu list. This tabu search algorithm stops after the maximum number of rejections has been surpassed. At that time the best routing is revealed.

D. SIMULATED ANNEALING/TABU SEARCH (SA/TABU)

The previous algorithms have been studied in detail on a number of different problems. The next algorithm is a new approach to solving the Vehicle Routing Problem. It has been mentioned in previous literature that Tabu Search can guide other solution methods in finding the global optimal values for specific problems [Glo 90]. What if Simulated Annealing were merged with Tabu Search? What would happen to the solution of the problem? Will giving Simulated Annealing "memory" help it find better solutions than those previously published in the literature? Thus was born the Simulated Annealing/Tabu Search hybrid.

The algorithm to run Simulated Annealing/Tabu Search is similar to the algorithm used in Simulated Annealing. Refer to Figure 5 to view the algorithm. The hybrid algorithm will incorporate the tabu list of length t (short term memory) from Tabu Search into the Simulated Annealing GENERATE function. As a result, Simulated Annealing will not be allowed to reuse a "move" until after t iterations have passed. This should lead Simulated Annealing to make better "move" selections than it did with no memory. This addition should keep Simulated Annealing from cycling for at least t iterations, also.

IV. EXPERIMENTATION AND RESULTS

The six problems solved by all three algorithms were the problems annotated in [Chr 79] and in [Chr 69]. The first four test cases contain cities randomly generated in the plane. The last two test cases contain cities appearing in clusters. The results are presented in each section. These results are then analyzed.

A. SIMULATED ANNEALING

Simulated Annealing is a highly parameterized algorithm. Parameters that can be varied are starting temperature, temperature decrement, Markov Chain length, and stopping criterion [AaV 85]. The starting temperature informs the algorithm of the temperature to which the "melt" or configuration was raised before "cooling" begins. The temperature decrement indicates how fast or slow the configuration will be "cooled." The Markov Chain length communicates the number of new configurations the algorithm is to generate before the temperature is allowed to decrease.

The parameters studied for this algorithm were the Markov Chain length and the temperature decrement. The starting temperature was held constant in these tests at 30 units. The stopping criterion implemented required every configuration to be rejected before Simulated Annealing would stop. These tests were performed on the fifty city problem [Chr 69]. Temperature decrement and Markov Chain length were allowed to

vary because previous literature indicates that these two parameters have a direct impact on solution quality. A proper mix of parameters is required to encourage Simulated Annealing to find a global optimal value.

If the temperature decrement is too fast, the configuration will be "cooled" too rapidly. Rapid "cooling" can lead to the algorithm becoming trapped in a local optimal valley instead of allowing the algorithm to "climb" out of that valley and continue searching for the global optimal value. If the temperature decrement is "cooled" too slowly, the algorithm may not converge in an acceptable time period.

If the Markov Chain length is too small, the algorithm will be restricted in its solution space exploration leading again to a suboptimal configuration. However, if the Markov Chain length is too long the resulting solution may also be suboptimal because the algorithm was allowed to "climb" out of the global optimal valley.

The Markov Chain length values tested were 100, 150, and 250. The temperature decrement values tested were 90%, 95%, and 99% of the previous temperature. In the testing process each parameter was held constant while the other parameter was varied. Since the algorithm creates a number of routings equal to the number of cities in the first route, the total route distances for each routing were added together and divided by the number of cities in the first route for an average configuration cost. This cost was then used in determining the best set of parameters.

From this testing, it appears that the best value for the Markov Chain length is 100 configurations and the best value for the temperature decrement is 99% of the

previous temperature. This combination of values produced the best known optimal total route distance for the fifty city problem.

Once these parameters were located, the next five problems were run with a Markov Chain length of 100 and a temperature decrement of 99% of the current temperature. The results are presented in Table I. These results were compared with Simulated Annealing by Osman [Osm 93]. Osman's algorithm was chosen for comparison because it uses Simulated Annealing as the heuristic search method.

As the results indicate, Osman's algorithm outperforms WSA, except for the first case. The parameters applied to WSA were the ones obtained from the testing on the fifty city problem. Obviously, Simulated Annealing is extremely problem dependent. One explanation for poor performance is that the parameters should be retested on each problem to find the best set of parameters for that specific problem.

Table I. Results of Simulated Annealing Algorithm

Problem	n	OSA	WSA
1	50	528	524.61
2	75	838.62	868.87
3	100	829.18	849.14
4	150	1058	1102.23
5	100	826	858.44
6	120	1176	1249.13

B. TABU SEARCH

Similar to Simulated Annealing, Tabu Search requires the proper mix of parameters. Parameters of this solution method that can be varied are the length of the tabu list, number of rejections, and number of "move" configurations. The number of rejections indicates the criterion for stopping after a "best" solution is found. "Move" configurations are generated before selecting a "best" or least costly move. This parameter informs the algorithm of how many "moves" should be generated. "Moves" are generated by choosing neighbors from route neighborhoods. These "moves" dictate which cities will be inserted into existing routes.

If the tabu list is too short or too long, solution quality will suffer. If the number of rejections is too low, the search may stop prematurely. Conversely, if the number of rejections is too large, computing time will be wasted rejecting non-optimal solutions. Finally, if the number of "moves" generated is too low, too few "moves" will be investigated thus reducing solution quality. Again, if the number of "moves" generated is too large, computing time will be wasted generating "moves."

For each test two parameters were held constant while one parameter was varied. Testing was again performed on the fifty city problem. Tests were performed with the values of 2, 5, 7, and 10 for tabu list length, 10n, 30n, and 50n number of rejections where n is the number of cities in the problem, and 5, 10, and 15 "move" generations. Similar to Simulated Annealing an average total route distance was used in finding the best parameters. From the tests conducted, a 10 element tabu list, 50n rejections, and 15 "move" generations produced the best distance. Once this was accomplished the

remaining five problems were run with these parameters. Table II presents the results obtained from the WTS runs.

Table II. Results of TABU Search Algorithm

Problem	n	OTS	PF	T	TR	WTS
1	50	524.61	536	524.61	524.61	524.61
2	75	844	842	835.32	835.32	870.50
3	100	835	851	828.98	826.14	832.62
4	150	1044.35	1081	1029.64	1031.07	1093.94
5	100	819.59	826	819.56	819.56	836.28
6	120	1042.11	1049	1073.05	1042.11	1196.54

As shown in Table II, WTS is outperformed by each algorithm again, except for the first and third problems. Tabu Search, like Simulated Annealing, appears to be problem dependent. Parameter testing should be performed to find the best parameters to solve each specific problem.

C. SIMULATED ANNEALING/TABU SEARCH (SA/TABU)

No parameter testing was performed on SA/TABU. Instead, the parameters from WTS and WSA were combined to run the six test problems using SA/TABU. These parameters were chosen on the supposition that they would produce near-optimal

results for SA/TABU since they produced the best results for WSA and WTS. This, however, was not the case, as shown in Table III.

SA/TABU did not perform as well as either WSA or WTS alone with the exception of WSA on problem 6. The SA/TABU algorithm is not totally without merit, though. As with WSA and WTS, the algorithm is definitely problem dependent. Again, parameter testing appears to be the only solution to obtaining results highly competitive with previously published results.

D. DISCUSSION

The three algorithms discussed in this thesis competed well against each other. WTS found the best solution values for four of the six test problems. WSA found better values than either WTS or SA/TABU on the other two test problems. Separate parameter tests, similar to those performed on WSA and WTS, could be conducted on the SA/TABU algorithm. This should produce solution values as good or better than WSA or WTS. All three algorithms could also be modified to use a different method of route creation. This change might also allow the algorithms to be more competitive with each other.

The algorithms presented in this thesis, though competitive against each other, were not as competitive against the other previously published algorithms as shown in Table III. The results seem to indicate that further parameter testing needs to be performed. This appears to hold true for heuristic research in general. For instance, one of the reported algorithms, TR, was tested on each problem with many different

parameters before the values shown in the table were acheived [Gen 92]. However, this extensive testing did not produce optimal values all of the time. Further study into the parameters for the algorithms may assist each algorithm in finding an optimal value for each problem.

Table III. Results of SA/TABU Algorithm

Problem	n	OSA	OTS	PF	T	TR	WSA	WTS	SA/TABU
1	50	528	524.61	536	524.61	524.61	524.61	524.61	529.17
2	75	838.62	844	842	835.32	835.32	868.87	870.50	870.55
3	100	829.18	835	851	828.98	826.14	849.14	832.62	849.83
4	150	1058	1044.35	1081	1029.64	1031.07	1102.23	1093.94	1102.23
5	100	826	819.59	826	819.56	819.56	858.44	836.28	860.41
6	120	1176	1042.11	1049	1073.05	1042.11	1249.13	1196.54	1201.29

V. CONCLUSIONS

The algorithms presented in this paper were Simulated Annealing, Tabu Search, and a Simulated Annealing/Tabu Search hybrid. These algorithms produced mixed results on the six test problems. However, the three algorithms competed well against each other, as previously stated. It appears that further parameter testing is required. Parameter testing appears to be a key element in producing optimal solution values from these heuristic solution methods. If the parameter tests are performed, the three algorithms might be highly competitive with each other.

The algorithms were competitive on two of the test problems. One reason for the lack of competition, again, is the parameters that were used. All three algorithms produced results utilizing the parameters obtained from the tests run on the fifty city problem. Again, this leads to the conclusion that extensive parameter testing and tuning should be performed. With regard to the published algorithms, it is known in some cases that extensive parameter tests were performed, while in other cases, the authors do not divulge their methods of experimentation. For example, the TR algorithm by Gendreau, Hertz, and Laporte did go through extensive parameter tests for each problem before the near optimal values were found [Gen 92]. Consequently, if better parameter values were obtained through thorough testing, the solution values produced by the three algorithms in this thesis would be improved.

Some future research that could be conducted entails implementing a different insertion algorithm in either WSA, WTS, or SA/TABU to obtain better results than presented here. An insertion method similar to the GENI algorithm in TR would be an excellent candidate. Also, each algorithm should be subjected to extensive parameter testing and tuning to produce optimal or near-optimal solution values. Finally, all the Tabu Search components could be added to WTS to investigate what effect diversification and intensification have on solution quality.

BIBLIOGRAPHY

[AaV 85] Aarts, E.H.L. and P.J.M. Van Laarhoven. "Statistical Cooling: A General Approach to Combinatorial Optimization Problems," Philips Journal of Research, 40(1985), 193-226.

[Chr 79] Christofides, Nicos, et.al. Combinatorial Optimization, Chichester: John Wiley and Sons, 1979.

[Chr 69] Christofides, N. and S. Eilon. "An Algorithm for the Vehicle-dispatching Problem," Operational Research Quarterly, XX, 3(1969), 309-318.

[ClW 64] Clarke, G. and Wright, J.W. "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points," Operations Research, XII, 4(1964), 568-581.

[DaR 59] Dantzig, G.B. and Ramser, J.H. "The Truck Dispatching Problem," Management Science, VI, 1(1959), 80-91.

[Gen 92] Gendreau, Michel., et. al. "A Tabu Search Heuristic for the Vehicle Routing Problem," Publication CRT-777 Centre de recherche sur les transports, University of Montreal, Montreal, Canada, November, 1992.

[GiM 74] Gillett, Billy E. and Leland R. Miller. "A Heuristic Algorithm for the Vehicle-Dispatch Problem," Operations Research, XXII, 2(1974), 340-349.

[Glo 90] Glover, Fred. "Tabu Search: A Tutorial," Interfaces, XX, 4(1990), 74-94.

[Glo 91] Glover, Fred., et al. "A User's Guide to Tabu Search," Technical Report. University of Colorado, Boulder, November, 1991.

[Kir 83] Kirkpatrick, S., et al. "Optimization by Simulated Annealing," Science, CCXX, 4598(1983), 671-680.

[Kro 72] Krolak, Patrick., et al. "A Man-Machine Approach Toward Solving the Generalized Truck-Dispatching Problem," Transportation Science, VI, 2(1972), 149-170.

[Lin 65] Lin, Shen. "Computer Solutions of the Traveling Salesman Problem," Bell System Technical Journal, XLIV, 10(1965), 2245-2269.

[Osm 93] Osman, Ibrahim Hassan. "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem," Annals of Operations Research, 41(1993), 421-451.

[ThP 89] Thompson, Paul M. and Harilaos N. Psaraftis. "Cyclic Transfer Algorithms for Multi-Vehicle Routing and Scheduling Problems," Working Paper Series 89-008. Leavey School of Business and Administration Santa Clara University, Santa Clara, April, 1989.

VITA

Jeffrey Dale White was born on August 16, 1968 in Kansas City, Missouri. After receiving his primary and secondary education in Beatrice, Nebraska, he studied at MidAmerica Nazarene College in Olathe, Kansas. Upon completion of the degree requirements at MidAmerica Nazarene College, he earned a Bachelor of Arts degree in Computer Science and Mathematics in May 1990, graduating summa cum laude. At MidAmerica he received the awards of Outstanding First Year Computer Science Student and Outstanding Computer Science Graduate. He currently is a member of the Phi Delta Lambda honor society for graduates of Nazarene institutions.

Following graduation, he began graduate studies at the University of Missouri-Rolla in September 1990. During his studies, he has held a Chancellor's Fellowship and a graduate teaching assistantship.

Currently, he is employed as a Computer Programmer by the Missouri Highway and Transportation Department located in Jefferson City, Missouri.